



No se permiten ni libros, ni apuntes, ni calculadoras programables. Sólo se permite el manual del microcontrolador
Se contestará sólo en el espacio reservado al efecto, pudiendo utilizar la cara posterior de la misma hoja.
No se pueden separar las hojas ni presentar hojas adicionales. Todas las respuestas deben estar justificadas.

APELLIDOS	NOMBRE	NIA
	<i>SOLUCIÓN</i>	

CUESTIÓN 1 (2.5 puntos):

a) (1.5 puntos) Se quiere que un STM32L152RB conectado a una placa STM32L_Discovery, con `clk1` de 16MHz, genere una señal de periodo 1ms, con ciclo de trabajo variable entre 10% y 90%. La señal deberá salir por el pin PB7. Realice la configuración inicial necesaria para empezar la generación de señal con un 10%.

Se trata de generar una señal PWM que salga por PB7, el cual está vinculado al canal 2 del TIM4. Por lo tanto la configuración será como sigue:

```
// PB7 como salida de TIM4
GPIOB->MODER|=0x00000001 << (2*7 +1);
GPIOB->MODER&=~(0x00000001 << (2*7));
GPIOB->AFR[0]=(0x02 << (7*4));
// Selección de reloj interno
TIM4->CR1 = 0x0080; // Autoreload activado, contador deshabilitado
TIM4->CR2 = 0x0000; // Como se ha comentado en clase, se deja a 0
TIM4->SMCR = 0x0000; // Como se ha comentado, también se deja a 0
TIM4->PSC = 159; // Contador en medidas de 0,01ms @ 16MHz
// Valores en ARR y CCRx
TIM4->CNT = 0;
TIM4->ARR = 99; // Reinicio a las 100 cuentas (a 1ms)
TIM4->CCR2 = 9; // pulso del 10%
// IRQ
TIM4->DCR = 0; // Sin DMA
TIM4->DIER = 0x0000; // Sin interrupciones
// Modo de salida
TIM4->CCMR1 = 0x6800; // CH2: Funcionalidad TOC y PWM que empieza en '1' y preload
TIM4->CCMR2 = 0x0000; // Sin configurar CH3 y CH4
TIM4->CCER = 0x0010; // Activada la salida hardware del canal 2
// Habilitación de contador
TIM4->EGR |= 0x0001; // Se actualizan los registros internos
TIM4->CR1 |= 0x0001; // Habilitación del contador
TIM4->SR = 0; // Se limpian los flags
```



UNIVERSIDAD CARLOS III DE MADRID
Sist. Dig. Basados en Microprocesador
8 de mayo de 2012

(Dpto. de Tecnología Electrónica)
(Gr. Ing. Telemática)
2º PARCIAL (75 minutos)

*No se permiten ni libros, ni apuntes, ni calculadoras programables. Sólo se permite el manual del microcontrolador
Se contestará sólo en el espacio reservado al efecto, pudiendo utilizar la cara posterior de la misma hoja.
No se pueden separar las hojas ni presentar hojas adicionales. Todas las respuestas deben estar justificadas.*

APELLIDOS	NOMBRE	NIA
	SOLUCIÓN	

b) (1.0 puntos) Ahora escriba las sentencias necesarias para que la señal generada cambie a tener un ciclo de trabajo del 90%.

Mediante el uso de la funcionalidad de PWM del microcontrolador, una vez configurado el temporizador, lo único que hay que hacer es cambiar el valor del tiempo que va a estar el pulso a nivel alto, es decir, cambiar el CCR2:

TIM4->CCR2 = 89;



CUESTIÓN 2 (2.5 puntos):

Para una aplicación determinada basada en un STM32L152RB y una placa STM32L_Discovery, funcionando con un pclk de 24MHz, se ha generado el siguiente código, el cual puede tener errores y/o ausencia de código.

```
1. #include "stm32l1xx.h"
2. #include "..\Biblioteca_SDM.h"
3.
4. int main(void){
5.     Init_SDM();
6.     GPIOB->MODER |= (0x01 << (2*9+1));
7.     GPIOB->MODER &= ~(0x01 << (2*9));
8.     GPIOB->MODER |= (0x01 << (2*8+1));
9.     GPIOB->MODER &= ~(0x01 << (2*8));
10.    GPIOB->AFR[0] = 0x77000000;
11.    USART1->CR1 = 0x0000000C;
12.    USART1->CR2 = 0x00000000;
13.    USART1->BRR = 0x00000068;
14.    USART1->CR1 |= 0x01 << 13;
15.    USART1->CR1 |= 0x01 << 5;
16.
17.    while (1) {
18.        while (valor==0);
19.        while ((USART1->SR & 0x0080)== 0);
20.        USART1->DR = valor;
21.        valor=0;
22.    }
23. }
```

Respecto a ese código conteste a las siguientes preguntas:

a) (1,0 puntos) ¿Qué periférico se está utilizando? ¿Cómo está configurado?

La USART1. Su configuración es de la siguiente manera:

- *Según CR1 (línea 11): Oversampling a 16, deshabilitada, 8-bits, Sin paridad, Sin interrupciones, habilitada la Tx y la Rx*
- *Según CR2 (línea 12): 1 bit de parada*
- *Según BRR (línea 13): pre-escalado 6.5, que a 24MHz da un baudrate de 230.4Kbps*
- *Según la línea 14, se activa la USART*
- *Según la línea 15, se activa la interrupción por Rx*

Por lo tanto su configuración es a 230400,8,N,1, activada tanto en recepción como en transmisión, y con la recepción por interrupciones.



b) (0,5 puntos) ¿Hay algún error en el código? Si es así, soluciónelo indicando las líneas donde se encuentra dicho error

Hay varios errores, muchos de ellos por falta de código (que se contestará en el apartado siguiente) y otros por fallos en el código escrito.

El error debido al código escrito, es que si bien la USART está completamente configurada, los pines por donde salen las señales de Tx y de Rx no están bien configurados, puesto que deben ser los pines 6, en lugar de 8, y 7 en lugar de 9. Por lo tanto las líneas 6-9 deben modificarse, sustituyendo los 8 por 6 y los 9 por 7.

c) (1,0 puntos) ¿Faltan líneas de código? Si es así, añádalas, indicando entre qué líneas las incluiría.

Hay 3 ausencias:

- *Falta la rutina de atención a la interrupción de la Rx de la USART, que debe recoger el byte recibido y meterlo en la variable valor. Por lo tanto, antes de la línea 4 habría que escribir dicha RAI:*
 - `void USART1_IRQHandler(void) {`
 - `if ((USART1->SR & 0x0020)!=0) {`
 - `valor = USART1->DR;`
 - `}`
 - `}`
- *Falta la definición de la variable valor, que ha de hacerse global, por lo tanto antes de la RAI anteriormente mencionada*
 - `unsigned char valor;`
- *Falta la activación del NVIC para que esté activada la interrupción de la USART1. Habría que ponerlo después de la activación de la interrupción en la USART, por lo tanto después de la línea 15*
 - `NVIC->ISER[1] |= (1 << (37-32));`



CUESTIÓN 3 (2.5 puntos):

Respecto al lenguaje ensamblador de un ARM, complete la siguiente tabla (cada fila vale 0,2+0,2+0,1 puntos):

INSTRUCCIÓN	MODO DE DIRECCIONAMIENTO	DESCRIPCIÓN EN RTL	REGISTROS QUE CAMBIAN SU VALOR
ADDS R2, R1, #0x0800	<i>Inmediato</i>	$R2 \leftarrow R1 + 0x0800$	<i>R2</i>
MOVS R2, R6 LSL R1	<i>Directo a registro con desplazamiento</i>	$R2 \leftarrow R6 \ll (\text{tantas veces como diga } R1)$	<i>R2</i>
LDR R5, [R2, -R6]	<i>Indexado</i>	$R5 \leftarrow (R2 - R6)$	<i>R5</i>
STR R3, [R2], #53	<i>Indirecto con postincremento</i>	$(R2) \leftarrow R3$ $R2 \leftarrow R2 + 53$	<i>R3 y R2</i>
STR R8, [R2, #3]!	<i>Indirecto con pre-incremento</i>	$R2 \leftarrow R2 + 3$ $(R2) \leftarrow R8$	<i>R2 y R8</i>



CUESTIÓN 4 (2.5 puntos):

Se tiene un microprocesador con las siguientes características:

- Arquitectura von Neumann
- 8 registros internos que pueden ser utilizados tanto para direcciones como para datos, sin poder acceder a la memoria de programa.
- Memoria de 1KB organizados en palabras de 8 bits
- Filosofía Load & Store

a) (1,5 puntos) Diseñe una codificación de los distintos tipos de instrucciones microprocesador, minimizando en lo posible el tamaño de la palabra de instrucción. Los tipos de instrucciones que debe tener el microprocesador, son:

- 10 instrucciones aritmético/lógicas de operar entre registros, contemplando que el resultado siempre se guarda en uno de los dos registros de los operandos
 - Ejemplo: ADD Rd, Rs; $Rd \leftarrow Rd + Rs$
- 7 instrucciones aritmético/lógicas con direccionamiento inmediato, donde el resultado se almacenará en el registro del otro operando.
 - Ejemplo: ADDI Rd, 115; $Rd \leftarrow Rd + 115$
- 5 instrucciones de salto absoluto condicional
 - Ejemplo: JMP <direccion>; $PC \leftarrow \langle \text{direccion} \rangle$
- 4 instrucciones de transferencia de información, con direccionamiento directo
 - Ejemplo: LD Rd, <direccion>; $Rd \leftarrow \langle \text{direccion} \rangle$
- 7 instrucciones de transferencia de información, con direccionamiento indirecto
 - Ejemplo: LDI Rd, Ri; $Rd \leftarrow (Ri)$

Para direccionar una memoria de 1KB se necesitan 10 líneas de dirección (10 bits).

Para direccionar los registros, como son 8, se necesitan 3 bits.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Tipo 1	0	0	0	0	X	X	opcode				Rs			Rd			
Tipo 2	0	1	opcode			dato							Rd				
Tipo 3	0	0	1	opcode			dirección										
Tipo 4	1	opcode			Rd			dirección									
Tipo 5	0	0	0	1	X	X	X	opcode				Ri			Rd		

Como se puede ver, las instrucciones deberían ser de 16 bits (2 palabras).



b) (1,0 puntos) Indique el tamaño de:

a) Registro de Instrucción

Como las instrucciones son de 16 bits, el IR también debe ser de 16 bits.

b) Contador de Programa

Como la memoria es de 1K instrucciones, el PC debe ser de 10 bits.

c) Registros internos

Como los registros sólo usan datos (8bits), o direcciones de datos (10bits), su tamaño tiene que ser el máximo entre esos dos parámetros. Por lo tanto serán de 10 bits.